Honeywell Bull

TIME-SHARING APPLICATIONS LIBRARY GUIDE VOLUME I - MATHEMATICS ADDENDUM A

SERIES 600/6000

APPLICATIONS

SUBJECT:

Additions to the Mathematical Time-Sharing Programs.

SPECIAL INSTRUCTIONS:

This update, Order Number DA43A, is the first addendum to DA43, Revision 0, dated June 1971. The attached pages are to be inserted into the manual as indicated in the collating instructions on the back of this cover.

Ten additional programs are being added with this addendum; these programs are listed in the Preface.

NOTE: This cover should be inserted following the manual cover to indicate the updating of the document with Addendum A.

DATE:

December 1972

ORDER NUMBER:

DA43A, Rev. 0

6334
1.5173
Printed in France

Ref.: 19.53.106 A1

COLLATING INSTRUCTIONS

To update this manual, remove old pages and insert new pages as follows:

Remove	Insert
Catalog	12/72 Preface
	Catalog
	*BASE (after ARCTAN)
	*CPOLY (after COMP3)
	*CPOLY-DR
	*EIGNHC (after EIGi)
	*EIGNSR
	*LINSD (after LINEQ)
	*LINSS (after LINSR)
	*MTRAN (after MTMPY)
	*SPLINE (after SPEIGI)
	*SPLINT

Programs preceded by an asterisk (*) represent new programs that are being added at this time; be certain to write these into the Table of Contents for future reference.

©1973, Honeywell Information Systems Inc.

File No.: 16M3, 17M3, 1ZM3

PREFACE

This manual describes and discusses the usage of the mathematical time-sharing programs available with Series 600 and 6000 information processing systems. The programs are listed alphabetically in the table of Contents.

Each program description includes the purpose of the program; language in which it is written; method of approach, if applicable; instructions for use; restrictions if any; and sample problems and solutions. In the sample solutions, all information typed by the user is underlined.

The instructions provided assume that the programs are available in the user master catalog LIBRARY and accessible with READ or EXECUTE permission. In the sample solution printouts, the programs had already been accessed using the GET command, and/or copied onto the current file using the OLD or LIB command.

Time-sharing programs for other classifications are also available from Honeywell under the following titles:

Series 600/6000 Time-Sharing Applications Library Guide, Volume II - Statistics, Order No. DA44

Series 600/6000 Time-Sharing Applications Library Guide, Volume III - Industry, Order No. DA45

Series 600/6000 Time-Sharing Applications Library Guide, Volume IV - Business and Finance, Order No. DA46

A complete listing of the programs in the library is available by listing the Library program, CATALOG. A copy of this listing follows the table of Contents for your information.

ADDITIONAL PROGRAMS INCORPORATED IN THE DECEMBER 1972 ADDENDUM

BASE
CPOLY
CPOLY-DR
EIGNHC
EIGNSR
LINSD
LINSS
MTRAN
SPLINE
SPLINT

All programs listed above (with the exception of SPLINE and SPLINT) have been coded and tested using Fortran Y. When using these particular programs, enter YFOR after the terminal inquires 'SYSTEM?".

Series 600/6000 Time Sharing Applications Library programs are available to users of the DATANETWORK service. Please contact your local Honeywell representative for further details.

This document describes programs that originated from a variety of sources, such as users and the Honeywell field organization. The programs and documentation are made available in the general form and degree of completeness in which they were received. Honeywell Information Systems Inc., therefore, neither guarantees the accuracy of the programs nor assumes support responsibility.

CATALØG ØF SERIES 6000/600 T-S LIBRARY PRØGRAMS

FILE TYPE INDICATØR:

LANGUAGE (FIRST LETTER)		MØDE (FØLLØWING LETTERS)	
A	ALGØL	P	(ØR BLANK) PRØGRAM
B	BASIC	S	SUBRØUTINE(S)
C	CARDIN	F	FUNCTIØN(S)
D	DATABASIC	P-S	PRØGRAM WITH EXTRACTABLE SUBRØUTINE(S)
E	TEXT EDITØR	R	RELØCATABLE ØBJECT (C*)
F	FØRTRAN	H	SYSTEM LØADABLE ØBJECT (H*)

ALL FILES ARE SØURCE MØDE UNLESS ØTHERWISE INDICATED.

SUBJECTS	DØCUMENTATIØN MANUAL
MATHEMATICS (MA)	..... ØRDER # DA43
INTEGRATIØN	
DIFFERENTIATIØN, DIFFERENTIAL EQ.	
INTERPØLATIØN	
PØLYNØMIALS	
LINEAR EQUATIØNS	
MATRICES	
NØN-LINEAR EQUATIØNS	
SPECIAL FUNCTIØN EVALUATIØN	
LØGIC AND NUMBER THEØRY	
STATISTICS (ST)	..... ØRDER # DA44
CURVE FITTING AND REGRESSIØN	
ANALYSIS ØF VARIANCE	
PRØBABILITY DISTRIBUTIØNS	
CØNFIDENCE LIMITS	
HYPØTHESIS TESTING	
DESCRIPTIVE STATISTICS	
RANDØM NUMBER GENERATIØN	
MISCELLANEØUS STATISTICS	
BUSINESS AND FINANCE (BF)	..... ØRDER # DA46
MANAGEMENT SCIENCE AND ØPTIMIZATIØN (MS)	.... ØRDER # DA45
LINEAR PRØGRAMMING	
INTEGER PRØGRAMING	
NØN-LINEAR ØPTIMIZATIØN	
NETWØRK ANALYSIS	
FØRECASTING	
SIMULATIØN	
ENGINEERING (EN)	
GEØMETRIC AND PLØTTING (GP)	
EDUCATIØN AND TUTØRIAL (ED)	
DEMØNSTRATIØN (DE)	
UTILITY AND MISCELLANEØUS (UM)	

THE DØCUMENTATIØN FØR THESE PRØGRAMS IS AVAILABLE IN FØUR MANUALS:
SEE ØRDER # DA43 FØR PRØGRAMS IN MATHEMATICS
ØRDER # DA44 FØR PRØGRAMS IN STATISTICS
ØRDER # DA46 FØR PRØGRAMS IN BUSINESS AND FINANCE
ØRDER # DA45 FØR PRØGRAMS IN ALL ØTHER CATEGØRIES.

SUBRØUTINES THAT ARE CALLED BY A PRØGRAM AND MUST BE EXECUTED WITH IT ARE LISTED IN BRACKETS AT THE END ØF THE DESCRIPTIØN.

THESE PRØGRAMS HAVE ALL BEEN REVIEWED AND TESTED BUT NØ RESPØNSIBILITY CAN BE ASSUMED.

**********MA--MATHEMATICS**********

INTEGRATION

QLCINT	FF	INTEGRATION BY SIMPSON'S RULE
FINT	FF	EVALUATE FOURIER INTEGRALS BY FILON'S FORMULA
GAHER	FF	GAUSS-HERMITE QUADRATURE
GALA	FF	GAUSS-LAGUERRE QUADRATURE
GAUSSN	FF	EVALUATE DEFINITE DOUBLE OR TRIPLE INTEGRALS
GAUSSQ	FF	GAUSSIAN QUADRATURE
NCOATES	FP-S	NEWTON-COATES QUADRATURE
NUMINT	B	GAUSSIAN QUADRATURE
ROMBINT	FP-S	ROMBERG INTEGRATION
SPLINE	B	INTEGRATE TABULATED FUNCTION BY SPLINE FITS

DIFFERENTIATION, DIFFERENTIAL EQ.

AMPBX	FS	ADAMS-MOULTON FOR 1ST-ORDER DIFF. EQNS (RKPBX)
FDRVUL	FF	DIFFERENTIATE TABULATED FUNCTION, UNEQUAL SPACING
HDRVEB	FF	DIFFERENTIATE TABULATED FUNCTION, EQUAL SPACING
RKPBX	FS	RUNGE-KUTTA FOR 1ST-ORDER DIFF. EQNS

INTERPOLATION

SPLINT	B	SPLINE INTERPOLATION
TNT1	FF	SINGLE LAGRANGIAN INTERPOLATION (TLU1)
TNT2	FF	DOUBLE LAGRANGIAN INTERPOLATION (TLU1)
TNT2A	FF	VARIABLE DOUBLE LINEAR INTERPOLATION (TLU1)

POLYNOMIALS

BICOF	FS	CALCULATE BINOMIAL COEFFICIENTS
QLPLY	FF	EVALUATE REAL POLY AT REAL ARGUMENT
CPOLY	FS	FINDS ZEROES OF A COMPLEX POLYNOMIAL
CPOLY-DR	FP	FINDS ZEROES OF A COMPLEX POLYNOMIAL (CPOLY)
DVALG	FS	POLYNOMIAL DIVISION
EUALG	FS	G.C.D. OF TWO POLYNOMIALS (DVALG)
MTALG	FS	MULTIPLY POLYNOMIALS
PLMLT	FS	REAL POLY COEFFICIENTS RECONSTRUCTED FROM REAL ROOTS
POLRTS	FP	SOLUTION OF POLY BY BAIRSTOWS METHOD
POLYC	FS	REAL POLY COEFFICIENTS RECONSTRUCTED FROM COMPLEX ROOTS
POLYV	FS	EVALUATE REAL POLY AT COMPLEX ARGUMENT
QUADEQ	B	SOLUTION TO QUADRATIC EQUATIONS
ROOTER	B	SOLUTION OF POLY BY BAIRSTOWS METHOD
ZCOP	FP	ROOTS OF POLYNOMIAL WITH COMPLEX COEFF.
ZCOP2	FS	ROOTS OF POLYNOMIAL WITH COMPLEX COEF. (ZCOP2)
ZORP	FP	ROOTS OF REAL POLY
ZORP2	FS	ROOTS OF REAL POLY

LINEAR EQUATIONS

GJSIMEQ	FS	SOLVE LINEAR SYSTEMS BY GAUSS-JORDAN
GSEIDEL	FP-S	SOLVE LINEAR SYSTEMS BY GAUSS-SEIDEL
LINEQ	FS	SOLVE LINEAR SYSTEMS BY GAUSSIAN ELIMINATION
LINSR	FP	SOLVE LINEAR SYSTEMS BY GAUSSIAN ELIMINATION (LINEQ)
SIMEQN	B	SOLVE LINEAR SYSTEMS BY MATRIX INVERSION

MATRICES

DETE	FF	EVALUATE DETERMINANT OF REAL MATRIX
DOMEIG	FP-S	DOMINANT EIGENVALUES OF REAL MATRIX
EIG1	FS	EIGENVALUES OF SYM MATRIX BY JACOBI METHOD
EIGNHC	FS	EIGENVALUES & VECTORS OF COMPLEX NON-HERMITIAN MATRICES
EIGNSR	FS	EIGENVALUES & VECTORS OF REAL NON-SYMMETRIC MATRICES
EIGSR	FP	EIGENVALUES AND VECTORS OF REAL SYM. MATRIX (EIG1)
LINSD	FS	SOLVE LIN. SYS. W/ SYMMETRIC DOUBLE PREC. COEF. MATRIX
LINSS	FS	SOLVE LIN. SYS. W/ SYMMETRIC SINGLE PREC. COEF. MATRIX
MTINV	FS	MATRIX INVERSION BY PIVOTS
MTMPY	FS	MATRIX MULTIPLICATION
MTRAN	FS	TRANSPOSE A MATRIX
SPEIG1	FS	SPECIAL EIGEN PROBLEMS (EIG1)
SYMEIG	FP	EIGENVALUES OF SYM MATRIX BY JACOBI METHOD

```
***NØN-LINEAR EQUATIØNS***
BRØWN      FS     SØLN ØF SIMULTANEØUS SYSTEMS BY BRØWN METHØD
SECANT     FS     SØLN ØF SIMULTANEØUS SYSTEMS BY SECANT METHØD [MTINV]
SØLN       FF     ZERØ ØF AN ARBITRARY FUNCTIØN
ZERØES     B      ZERØ,MAX,MIN ØF FUNCTIØN

***SPECIAL FUNCTIØN EVALUATIØN***
ARCTAN     FF     ARCTANGENT IN RADIANS ØF Y/X
BESL       FS     BESSEL FUNCTIØN [GAMF]
CØMP1      FF     EVALUATES REAL HYPERBØLIC TRIG FUNCTIØNS
CØMP2      FS     CØMPLEX MULT. AND DIVISIØN
CØMP3      FS     EVALUATES VARIØUS FUNCTIØNS FØR CØMPLEX ARGUMENT [CØMP2]
ERRF       FF     ERRØR FUNCTIØN
ERRINV     FF     INVERSE ERRØR FUNCTIØN
FRESNL     FS     EVALUATES FRESNAL INTEGRALS
GAMF       FF     GAMMA FUNCTIØN
JACELF     FS     EVALUATES JACØBIAN ELLIPTIC FUNCTIØNS SN, CN, DN
ØRTHP      FF     EVALUATE ØRTHØGØNAL PØLYNØMIALS
STIRLING   FP-S   N FACTØRIAL BY STIRLINGS APPRØXIMATIØN
TMFCEV     B      EVALUATE DAMPED ØR UNDAMPED FØURIER SERIES

***LØGIC AND NUMBER THEØRY***
4SQRS      B      WRITES INTEGERS AS SUM ØF SQUARES ØF FØUR INTEGERS
BASE       FP     CØNVERTS NUMBERS FRØM ØNE BASE TØ ANØTHER
CØNCLUDE   B      DETERMINES LØGICAL CØNCLUSIØNS FRØM PRØPØSITIØNAL LØGIC
GCDN       FS     G.C.D. ØF N INTEGERS

**********ST--STATISTICS**********

***CURVE FITTING AND REGRESSIØN***
CFIT       FP     LEAST SQRS. PØLY. WITH RESTRAINTS
CURFIT     B      FITS SIX DIFFERENT CURVES BY LEAST SQRS
FØRIR      FP     LEAST SQUARES ESTIMATE ØF FINITE FØURIER SERIES MØDEL
FØURIER    B      CØEFF ØF FØURIER SERIES TØ APPRØX A FUNCTIØN
LINEFIT    FS     LEAST SQRS LINE
LINREG     B      LST.SQRS. BY LINEAR, EXPØNENTIAL, ØR PØWER FUNCTIØN
LSPCFP     FP     LEAST SQRS PØLYNØMIAL FIT
LSQMM      FS     GENERALIZED PØLY FIT BY LEAST SQRS ØR MIN-MAX
MREG1      FP     MULTIPLE LINEAR REGRESSIØN
MULFIT     B      MULTIPLE LINEAR FIT WITH TRANSFØRMATIØNS
ØRPØL      FP     LEAST SQRS FIT WITH ØRTHØGØNAL PØLYS
PØLFIT     B      LEAST SQRS PØLYNØMIAL FIT
PØLFT      FP     LEAST SQRS PØLYNØMIAL FIT
SMLRP      FP     MULTIPLE LINEAR REGRESSIØN
SMLRPØBJ   FHP    SYSTEM LØADABLE FILE FØR SMLRP
STAT20     B      EFFRØYMSØN'S MULTIPLE LINEAR REGRESSIØN ALGØRITHM
STAT21     B      CØMPUTES MULTIPLE LINEAR REGRESSIØNS

***ANALYSIS ØF VARIANCE***
ANØVA      FP     ØNE ØR TWØ WAY ANALYSIS ØF VARIANCE
ANVA1      FP     ØNEWAY ANALYSIS ØF VARIANCE
ANVA3      FP     THREE WAY ANALYSIS ØF VARIANCE
ANVA5      FP     MULTIPLE VARIANCE ANALYSIS
KRUWAL     FP     KRUSKAL-WALLIS 2-WAY VARIANCE [XYNGAM]
ØNEWAY     B      ØNEWAY ANALYSIS ØF VARIANCE
STAT13     B      ANALYSIS ØF VARIANCE TABLE, 1-WAY RANDØM DESIGN
STAT14     B      ANALYSIS ØF VARIANCE TABLE FØR RANDØMIZED BLØCK DESIGN
STAT15     B      ANALYSIS ØF VARIANCE TABLE FØR SIMPLE LATIN-SQ DESIGN
STAT16     B      ANALYSIS ØF VARIANCE TABLE, GRAECØ-LATIN SQUARE DESIGN
STAT17     B      ANØVA TABLE ØF BALANCED INCØMPLETE BLØCK DESIGN
STAT18     B      ANALYSIS ØF VARIANCE TABLE, YØUDEN SQUARE DESIGN
STAT33     B      ANALYSIS ØF VARIANCE TABLE, 1-WAY RANDØM DESIGN
```

```
***PROBABILITY DISTRIBUTIONS***
ANPF        FF     NORMAL PROBABILITY FUNCTION (ERRF)
BETA        FF     BETA DISTRIBUTION
BINDIS      B      BINOMIAL PROBABILITIES
EXPLIM      B      EXPONENTIAL DISTRIBUTIONS
POISON      FF     POISSON DISTRIBUTION FUNCTION
PROBC       FP     PROBABLITIES OF COMBINATIONS OF RANDOM VARIABLES
PROVAR      B      NORMAL AND T-DISTRIBUTION
TDIST       FF     T-DISTRIBUTION (BETA)
XINGAM      FF     INCOMPLETE GAMA FUNCTION

***CONFIDENCE LIMITS***
BAYES       B      DIFFERENCE OF MEANS IN NON-EQUAL VARIANCE
BICONF      B      CONF. LIMITS FOR POPULATION PROPORTION (BINOMIAL)
BINOM       FP     BINOMIAL PROBABILITIES AND CONFIDENCE BANDS
COLINR      B      CONFIDENCE LIMITS ON LINEAR REGRESSIONS
CONBIN      B      CONF. LIMITS FOR POPULATION PROPORTION (NORMAL)
CONDIF      B      DIFFERENCE OF MEANS IN EQUAL VARIANCE
CONLIM      B      CONF. LIMITS FOR A SAMPLE MEAN
STAT05      B      CONFIDENCE INTERVAL FOR MEAN BY SIGN TEST
STAT06      B      CONFIDENCE LIMITS, WILCOXON SIGNED RANK SUM TEST

***HYPOTHESIS TESTING***
BITEST      B      TEST OF BINOMIAL PROPORTIONS
CHISQR      FS     CHI-SQUARE CALCULATIONS
CORREL      FP     CONTINGENCY COEFFICIENT (XINGAM)
CORRL2      FP     CORRELATION COEFFICIENT (TDIST;BETA)
KOLKO       FP     KOLMOGOROV-SMIRNOV TWO SAMPLE TEST (XINGAM)
SEVPRO      B      CHI-SQUARE
STAT01      B      MEAN,STD OF MEAN, ... , T-RATIO,2 GROUPS, PAIRED
STAT02      B      MEANS, VARIANCES, AND T-RATIO 2 GROUPS, UNPAIRED DATA
STAT04      B      CHI-SQUARE AND PROBABILITIES, 2X2 TABLES
STAT08      B      COMPARES TWO GROUPS OF DATA USING THE MEDIAN TEST
STAT09      B      COMPARE 2 DATA GROUPS,MANN-WHITNEY 2-SAMPLE RANK TEST
STAT11      B      SPEARMAN RANK CORRELATION COEF. FOR 2 SERIES OF DATA
STAT12      B      COMPUTES CORRELATION MATRIX FOR N SERIES OF DATA
TAU         FP     KENDALL-RANK CORRELATION

***DESCRIPTIVE STATISTICS***
MANDSD      B      FIND MEAN, VARIANCE, STD
STAT        FP     FIND SEVERAL STATISTICS FOR SAMPLE DATA (ANPF;ERRF)
STATAN      B      FIND VARIOUS STATISTICAL MEASURES
TESTUD      B      SAMPLE STATISTICS
UNISTA      B      DESCRIPTION OF UNI-VARIANT DATA

***RANDOM NUMBER GENERATION***
FLATSORC    C      CARDIN SOURCE FILE FOR FLAT
FLAT        FRF    UNIFORM RANDOM NUMBER GENERATOR
RANDX       FF     RANDOM #'S, UNIFORM DIST. BETWEEN 0 AND 1
RNDNRM      FF     CALCULATES NORMAL RANDOM NUM. (FLAT)
UNIFM       FRF    UNIFORM RANDOM NUMBER GENERATOR
UNIFMSOR    C      CARDIN SOURCE FILE FOR UNIFM
URAN        FRF    UNIFORM RANDOM NUMBER GENERATOR
URANSORC    C      CARDIN SOURCE FILE FOR URAN
XNORI       FF     NORMAL RANDOM NUMBERS, VARIABLE MEAN, STD (RANDX)
XNORM       FF     NORMAL RANDOM NUMBERS, MEAN 0, STD 1. (RANDX)

***MISCELLANEOUS STATISTICS***
FACTAN      FP     FACTOR ANALYSIS
STADES             EXPLANATION OF COLINR,CURFIT,MULFIT,UNISTA
```

```
**********BF--BUSINESS AND FINANCE**********

ANNUIT     B      ANNUITIES,LOANS,MORTGAGES
BLDGCOST   B      ANALYZE BUILDING COSTS
BONDATA    B      ANALYSIS OF A BOND INVESTMENT PORTFOLIO

BONDPR     B      COMPUTES PRICE AND ACCRUED INTEREST OF A BOND
BONDSW     B      CALCULATES THE EFFECT OF A BOND SWITCH
BONDYD     B      COMPUTES BOND YIELDS
CASHFLOW   B      PREDICTS NEXT YEARS CASH FLOW
DEPREC     B      CALCULATES DEPRECIATION BY FOUR METHODS
INSTLO     B      CALCULATES MONTHLY PAYMENT SCHEDULE ON INSTALLMENT LOAN
INVANL     FP     RETURN ON INVESTMENT ANALYSIS
LESSEE     B      COMPARES A LEASE WITH PURCHASE OF EQUIPMENT
LESSIM     B      SIMULATES LESSOR'S CASH FLOW AND RATE OF RETURN
LESSOR     B      CALCULATES THE LESSORS CASH FLOW & RATE OF RETURN
MAKE-BUY   B      TO MAKE OR TO BUY DECISIONS
MGSIM      FHP    SIMULATES COMPETITIVE INTERACTION OF COMPANIES
MGSIM-CS   FRP    OBJECT DECKS FOR MGSIM
MGSIM-IN          ON LINE INSTRUCTIONS FOR MGSIM
MORTCST    B      MORTGAGE SCHEDULE FOR VARIOUS TERMS
MORTGAGE   FP     CALCULATES A MORTGAGE REPAYMENT SCHEDULE
RETURN     B      COMPUTES ANNUAL RETURNS FOR A SECURITY FROM ANNUAL DATA
SALDATA    B      COMPUTES PROFITABILITY OF DEPARTMENTS OF A FIRM
SAVING     B      SAVINGS PLAN CALCULATIONS
SMLBUS     B      PAYMENT SCHEDULES FOR A SMALL BUSINESS ADMST. LOAN
TRUINT     B      INTEREST RATE CALCULATIONS

**********MS--MANAGEMENT SCIENCE AND OPTIMIZATION**********

***LINEAR PROGRAMMING***
ASSIGNIT   B      THE ASSIGNMENT PROBLEM
LINPRO     B      LINEAR PROGRAMMING
LNPROG     FP     LINEAR PROGRAMMING
TRANSPO    B      THE TRANSPORTATION PROBLEM
UNDEQ      FS     FINDS A SOLUTION FOR AN UNDERDETERMINED LINEAR SYSTEM

***INTEGER PROGRAMMING***
INTOI      FP     ZIONTS' MODIFICATION OF BALAS' ZERO-ONE ALGORITHM
INTLP      FP     GOMORY'S PURE AND MIXED INTEGER PROGRAMMING

***NON-LINEAR OPTIMIZATION***
CSM        FS     OPTIMIZE A LINEARLY CONSTRAINED CONVEX FUNCTION(UNDEQ)
DAVIDON    B      DAVIDON'S UNCONSTRAINED OPTIMIZATION
GEOSIM     B      HEURISTIC SCHEDULING OF N JOBS IN A M MACHINE SHOP
GPROG      FHP    SOLVES GEOMETRIC PROGRAMMING PROBLEMS
GPROG-SO   C      CARDIN SOURCE FILE FOR GPROG (UNDEQ,CSM)
JSSIM      B      SCHEDULES N JOBS IN A SHOP WITH M MACHINES
LOGIC3     FP     UNCONSTRAINED OPTIMIZATION
MAXOPT     FP     UNCONSTRAINED OPTIMIZATION

***NETWORK ANALYSIS***
CPM        FP     CRITICAL PATH METHOD
KILTER     FP     'OUT OF KILTER' ALGORITHM (MINIMUM COST CIRCULATION)
MAXFLOW    FP     MAXIMUM FLOW THRU NETWORK
PERT       B      SIMPLE ANALYSIS OF A PERT NETWORK
SHORTEST   FP     SHORTEST PATH - MIN SPANNING TREE

***FORECASTING***
COEFS      B      DETERMINE SEASONAL COEFFICIENTS ON TWO CYCLES
COMBI      B      DETERMINES ECONOMIC ORDER QUANITY FOR INVENTORY ITEMS
OPTIM      F      OPTIMUM SERVICE LEVEL FOR ONE INVENTORY ITEM
TCAST      FP     TIME SERIES FORECASTING (TCAST1,TCAST2)
TCAST1     FH     OVERLAY MODULE OF TCAST
TCAST      FHP    TIME SERIES FORECASTING
TCAST2     FH     OVERLAY MODULE OF TCAST
TCAST1     FH     OVERLAY MODULE OF TCAST
SMOOTH     FS     TRIPLE SMOOTHING OF A TIME SERIES
```

```
***SIMULATION***
GASPDATA  E      DATA FILE FOR SAMPLE PROGRAM GASPSAMP
GASPIIA   FS     'GASP' SIMULATION SYSTEM
GASPSAMP  FP     SAMPLE PROGRAM FOR GASPIIA [GASPIIA;GASPDATA]

********************EN--ENGINEERING************************************

ACNET     FP     FREQUENCY RESPONSE OF A LINEAR CIRCUIT
BEMDES    B      STEEL BEAM SELECTION
GCVSIZ    B      GAS CONTROL VALVE COEFF.
LCVSIC    B      LIQUID CONTROL VALVE COEFF.
LFILTR    B      SYNTHESIZES ACTIVE LOW-PASS FILTERS [LFLDAT]
LFLDAT           DATA FOR LFILTR
LFLTIN           INSTRUCTIONS FOR LFILTR
LPFILT    B      DESIGN LOW PASS FILTERS
NLNET     FP     GENERAL STEADY-STATE CIRCUIT ANALYSIS
OTTO      B      OTTO CYCLE OF ENGINE
PAVEIT    B      CALCULATES $ COST AND TONS OF MATERIAL TO PAVE A ROAD
PVT       FP     FINDS MOLAR VOLUME OF A GAS GIVEN TEMPERATURE AND PRES.
SCVSIZ    B      STEAM CONTROL VALVE COEFF.
SECAP     B      STEEL SECTION CAPACITIES

********************GP-GEOMETRIC AND PLOTTING***************************

CIRCLE    B      DIVIDES A CIRCLE INTO N EQUAL PARTS
PLOT      FS     PLOTS UP TO 9 CURVES SIMULTANEOUSLY
PLOTTO    B      SIMULTANEOUSLY PLOTS 1 TO 6 FUNCTIONS
POLPLO    FP     PLOTS EQNS IN POLAR COORDINATES
SPHERE    B      SOLVES ANY SPHERICAL TRIANGLE
TRIANG    B      SOLVES FOR ALL PARTS OF ANY TRIANGLE
TWOPLO    B      SIMULTANEOUSLY PLOTS 2 FUNCTIONS
XYPLOT    B      PLOTS SINGLE-VALVED FUNCTIONS

********************ED--EDUCATION AND TUTORIAL**************************

DRIVES    FHP    DRIVER FOR EXPER. A COMPUTER ASSISTED INST. LANG.
EXPERN    E      EXPER TUTORIALS IN EXPER (N=1 TO 5) [PREPRS;DRIVES]
PREPRS    FHP    PREPROCESSOR FOR EXPER. A COMPUTER ASSISTED INST. LANG.

********************DE--DEMONSTRATION**********************************

AMAZE     B      CONSTRUCTS MAZES    -  EACH UNIQUE
BLKJAK    B      THE COMPUTER DEALS BLACKJACK
POPING    B      POPULATION PROJECTIONS FOR AN AREA
PRIME     B      PRIME FACTORIZATION OF A NUMBER
XMAS      B      A HOLIDAY SING-ALONG, CHRISTMAS CARD AND GREETINGS

********************UM--UTILITY AND MISCELLANEOUS**********************

ADATER    FP-S   A CALENDER DATING ROUTINE
CATALOG   E      CATALOG OF SERIES 6000/600 T/S LIBRARY (THIS FILE)
CONVRT    B      CONVERTS MEASUREMENTS FROM ONE SCALE TO ANOTHER
DBLSORT   FS     SORT TWO ARRAYS
DESEQ     FP     STRIPS LINE SEQUENCE NUMBERS FROM A FILE
REFORM    FP     REFORMATS A 'NFORM' FORTRAN SOURCE FILE TO 'FORM'
RLINE     FS     READS LINE, OPTIONALLY STRIPS LINE # & COUNTS ENTRIES
SQLSORT   FS     SORT AN ARRAY
TLUI      FS     TABLE SEARCH
TPLSORT   FS     SORT THREE ARRAYS

***END OF CATALOG***
```

This Fortran program converts numbers from one base to another.

INSTRUCTIONS

After the computer requests an input line by typing an equal sign (=), enter

BIN, BOUT, DIG, NUM

separated by commas, where

BIN is the base of the input number.

BOUT is the base to which the number is to be converted.

DIG is the number of digits in the fractional part of the output number to be printed, if applicable.

NUM is the number to be converted. If the input base is greater than 10, separate the digits with blanks, and precede the "decimal point" with a blank.

If a base is set to 0, the number will be interpreted as a floating-point number expressed in octal format. If a base is set to -1, the number will be input or output in Fortran E format.

Alternate forms of input are:

BIN, BOUT, NUM

or

DIG, NUM

or

NUM

In these cases, the previously defined values of the deleted parameters are used.

The program will continue asking for new input until a null response is given.

NOTE: The program does not check to see if the digits of the input number are consistent with the input base.

SAMPLE PROBLEM

Convert 483 and 77241 from base 10 to base 8. Convert 614.35 from base 10 to base 8, carrying the answer to the ten-thousandths place. Convert (12)(3) 14).(11)(1) from base 16 to base 8, printing 10 digits in the fractional part. Then convert 140200 from Fortran E format to the floating-point octal format and then back to Fortran E format.

SAMPLE SOLUTION

```
*RUN
ENTER ? FOR INSTRUCTIONS

=10,8,483
( 8)743

=77241
( 8)226671

=4,614.35
( 8)1146.2631

=16,8,10,12 3 14 . 11 1
( 8)6076.5420000000

=-1,0,140200.
( 0)044421650000

=0,-1,044421650000
(-1)  0.14020000E 06

=

*
```

This Fortran subroutine finds the 0's of a complex polynomial.

REFERENCES

This algorithm was originally published as "Algorithm 419, " Communications of the ACM, Vol. 15, February 1972, page 97. It is reprinted here by permission of the Association for Computing Machinery.

METHOD

The routine finds the 0's of a complex polynomial one at a time in roughly increasing order of modulus and deflates the polynomial to one of lower degree. It uses the three stage algorithm of Jenkins and Traub. The timing is quite insensitive to the distribution of 0's.

INSTRUCTIONS

The calling sequence is

CALL CPOLY(OPR,OPI, DEGREE, ZEROR, ZEROI, FAIL)

OPR,OPI	double-precision vectors of read and imaginary parts of the coefficients in order of decreasing powers.
DEGREE	integer degree of polynomial
ZEROR, ZEROI	output double-precision vectors of real and imaginary parts of the 0's.
FAIL	output logical parameter, true only if leading coefficient is 0 or if the routine has found fewer than DEGREE 0's.

RESTRICTIONS

The algorithm will accept polynomials of maximum degree 49.

SAMPLE PROBLEM

Find the roots of the complex polynomial

$$X^5 + (3+i)X^4 + (3+i)X^3 + (3+i)X^2 + (3+i)X + (2+i)$$

SAMPLE SOLUTION

The following is a driver program written to solve the sample problem. The results follow:

```
*LIST

10 LOGICAL FAIL
20 DOUBLE PRECISION OPR(6),OPI(6),ZEROR(5),ZEROI(5)
30 DATA OPR/1.DO,4*3.DO,2.DO/,OPI/0.DO,5*1.DO/
40 CALL CPOLY(OPR,OPI,5,ZEROR,ZEROI,FAIL)
50 PRINT,"FAIL = ",FAIL
60 PRINT,"SOLUTION VECTOR"
70 PRINT 10,(ZEROR(I),ZEROI(I),I=1,5)
80 10 FORMAT(1X,2D25.18)
90 STOP
100 END


READY

*RUN *;CPOLY=(CORE=20)
FAIL =  F
SOLUTION VECTOR
 0.309016994374947424D 00 0.951056516295153572D 00
-0.809016994374947424D 00 0.587785252292473129D 00
-0.809016994374947424D 00-0.587785252292473128D 00
 0.309016994374947425D 00-0.951056516295153572D 00
-0.200000000000000000D 01-0.100000000000000000D 01

NORMAL TERMINATION

*
```

This Fortran program finds the 0's of a complex polynomial using the subroutine CPOLY.

INSTRUCTIONS

The library subroutine CPOLY must be referenced in the RUN list (see Sample Solution). Enter the complex coefficients in order of decreasing powers when requested by the program. The program will continue requesting additional polynomials until 0 is entered as the degree. The program will accept polynomials of maximum degree 49.

SAMPLE PROBLEM

Find the roots of the complex polynomial

$$X^5 + (3+i)\,X^4 + (3+i)\,X^3 + (3+i)X^2 + (3+i)X + (2+i)$$

CPOLY-DR-2

SAMPLE SOLUTION

```
*RUN CPOLY-DR;CPOLY = (CORE=20)

ENTER DEGREE = 0 TO STOP



DEGREE OF POLYNOMIAL
=5
 COEFFICIENTS IN ORDER OF DECREASING POWERS
REAL PART, IMAGINARY PART, ETC.
=1,0,  3,1,  3,1,  3,1,  3,1,  2,1.

SOLUTION VECTOR (CARTESIAN COORDINATES)
          REAL                          IMAGINARY

 3.09016994374947424D-01       9.51056516295153572D-01
-8.09016994374947424D-01       5.87785252292473129D-01
-8.09016994374947424D-01      -5.87785252292473128D-01
 3.09016994374947425D-01      -9.51056516295153572D-01
-2.00000000000000000D 00      -1.00000000000000000D 00

SOLUTION VECTOR (POLAR COORDINATES)
            R                           THETA

 1.00000000000000000D 00       7.19999999999999943D 01
 1.00000000000000000D 00       1.44000000000000004D 02
 1.00000000000000000D 00      -1.44000000000000004D 02
 1.00000000000000000D 00      -7.19999999999999943D 01
 2.23606797749978970D 00      -1.53434948822922017D 02



DEGREE OF POLYNOMIAL
=0

NORMAL TERMINATION
```

The Fortran subroutine, EIGNHC, finds the eigenvalues and eigenvectors of a complex non-Hermitian matrix. It also has the capability to determine a specified set of the eigenvectors.

INSTRUCTIONS

The calling sequence is:

CALL EIGNHC (A, NC, NV, ROOT, VECT, TEMP1, TEMP2, ITEMP3, IDIMA)

where,

- A is the complex matrix containing the eigenvalues and eigenvectors.
- NC is the order of the matrix, i.e., A is the NC by NC matrix.
- NV determines which vectors are found. If
  a. NV .GT. O, finds the eigenvectors associated with the NV eigenvalues of largest modulus.
  b. NV .LT. O, finds the eigenvectors associated with the IABS(NV) eigenvalues of smallest modulus.
  c. NV=O, no eigenvectors found.
- ROOT is the 1-dimensional complex array of the eigenvalues.
- VECT is the 2-dimensional complex array of the eigenvectors. The eigenvector corresponding to the I^{th} eigenvalue is stored columnwise in the I^{th} column of VECT.
- TEMP1 is a 2-dimensional complex array used internally. It is dimensioned as follows: TEMP1(IDIMA, NC).
- TEMP2 is a 2-dimensional complex array used internally. It is dimensioned as follows: TEMP2(IDIMA, 2).
- ITEMP3 is a 2-dimensional integer array used internally. It is dimensioned as follows: ITEMP3 (IDIMA, 2).
- IDIMA is the first dimension of the following arrays: A, VECT, TEMP1, TEMP2, ITEMP3.

RESTRICTIONS AND METHOD

1. After control is returned to the main program following a call to EIGRNS (EIGCNH), the routine may be entered again to find the eigenvectors of the transpose of A. This option must be used carefully, with the following restrictions

   a. The transpose of A must be stored before the first call to EIGNHC since the routine destroys A.

   b. The array ROOT must be the same in both calls to the routine since it contains the eigenvalues of A and, therefore, of A transpose: because of this, the second call to the routine does not recompute eigenvalues of A transpose. Therefore, no change can be made to the ROOT array before the second entry into EIGNHC.

   c. The reentry to the routine must be made with a minus NC in place of NC. This informs the routine that it already has the eigenvalues of A transpose which are stored in ROOT.

   d. Upon reentering, NV need not be the same as it was in the original entry.

This option provides the capability of solving the complete eigen problem, assuming the Jordan form D of A is diagonal. If X designates the matrix whose columns are eigenvectors of A, and Y designates the matrix whose columns are eigenvectors of A transpose, with proper scaling of X and Y, the following relations hold:

AX	=XD	(AT=A transpose, YT=Y transpose)
(AT)Y	=YD	
(YT)X	=I	(Scaling required to produce unit diagonal elements)
(YT)AX	=D	(The diagonal matrix D has the eigenvalues on the diagonal)

2. The eigenvectors returned by EIGNHC are normalized so that the largest component is 1.

3. It is suggested that the user check the modulus of the first component of each eigenvector after a call to EIGNHC. If the modulus is equal to 2., the routine has failed to determine the particular eigenvector. To determine the modulus use the CABS function.

REFERENCES

1. Francis, J.G.F., "The QR Transformation, A Unitary Analog to the LR Transformation," Part I, Computer Journal, Oct., 1961, 265-271.

2. _____, Part II, Computer Journal, Jan., 1962, 332-345.

3. Wilkinson, J. H., The Algebraic Eigenvalue Problem, Oxford Press, 1965.

4. G.E. TIS #67SD335, Eigenvalues and Eigenvectors of Non-Hermitian Matrices, by F. E. Lilley and A.T. Ross, G.E. Technical Information Exchange, P.O. Box 43, Bldg. 5; Schenectady, N.Y. 12301

SAMPLE PROBLEM

Given the matrix

$$A = \begin{bmatrix} 2-i & 0 & i \\ 0 & 1+i & 0 \\ i & 1-i & 2-i \end{bmatrix}$$

find all the eigenvalues and eigenvectors of A and A transpose.

SAMPLE SOLUTION

```
      PARAMETER MAX=10
      DIMENSION A(MAX,MAX),AA(MAX,MAX),ITEMP3(MAX,2)
      COMPLEX TEMP1(MAX,MAX),ROOT(MAX),TEMP2(MAX,2),VECT(MAX,MAX)
      COMPLEX VECT2(MAX,MAX),A,AA
    1 WRITE(6,1006)
      READ , NC
      IF(NC .LE. 0) STOP
      WRITE(6,1007)
      READ , NV
       WRITE(6,1008)
       READ , ((A(I,J),J=1,NC),I=1,NC)
       DO 10 I=1,NC
       DO 10 J=1,NC
   10 AA(J,I)=A(I,J)
       CALL EIGNHC(A,NC,NV,ROOT,VECT,TEMP1,TEMP2,ITEMP3,MAX)
       IF(NV)15,50,15
   15 CALL EIGNHC(AA,-NC,NV,ROOT,VECT2,TEMP1,TEMP2,ITEMP3,MAX)
       WRITE(6,1004)
       IF(NV)20,50,25
   20  K=NC+NV+1
       KK=NC
       GO TO 30
   25 K=1
       KK=NV
   30 DO 40 I=K,KK
       WRITE(6,1002)I,ROOT(I)
       WRITE(6,1009)I
   40 WRITE(6,1003)(VECT(J,I),J=1,NC)
       WRITE(6,1005)
       DO 45 I=K,KK
       WRITE(6,1002)I,ROOT(I)
       WRITE(6,1009)I
   45 WRITE(6,1003)(VECT2(J,I),J=1,NC)
       GO TO 1
   50 DO 60 I=1,NC
   60 WRITE(6,1002)I,ROOT(I)
       GO TO 1
 1002 FORMAT(/' EIGENVALUE',I3,':',2F12.6)
 1003 FORMAT(15X,2F12.6)
 1004 FORMAT(////25X,'MATRIX'/20X,'REAL',6X,'IMAGINARY')
 1005 FORMAT(////18X,'MATRIX TRANSPOSE'/20X,'REAL',6X,'IMAGINARY')
 1006 FORMAT(///'ENTER ORDER OF MATRIX')
 1007 FORMAT(/'ENTER NV CODE')
 1008 FORMAT(/'ENTER MATRIX BY ROWS')
 1009 FORMAT(' EIGENVECTOR',I3,':')
      END
```

```
*RUN *;EIGNHC=(CØRE=26)


ENTER ØRDER ØF MATRIX
=3

ENTER NV CØDE
=3

ENTER MATRIX BY RØWS
=2,-1,  0, 0,  0, 1
=0, 0,  1, 1,  0, 0
=0, 1,  1,-1,  2,-1
```

MATRIX

	REAL	IMAGINARY
EIGENVALUE 1:	2.000000	-2.000000
EIGENVECTØR 1:		
	-1.000000	-0.000000
	0.	0.
	1.000000	0.
EIGENVALUE 2:	2.000000	-0.000000
EIGENVECTØR 2:		
	1.000000	0.000000
	-0.000000	0.000000
	1.000000	0.
EIGENVALUE 3:	1.000000	1.000000
EIGENVECTØR 3:		
	-0.300000	0.100000
	1.000000	0.
	-0.700000	-0.100000

MATRIX TRANSPØSE

	REAL	IMAGINARY
EIGENVALUE 1:	2.000000	-2.000000
EIGENVECTØR 1:		
	-1.000000	0.
	0.400000	0.200000
	1.000000	0.
EIGENVALUE 2:	2.000000	-0.000000
EIGENVECTØR 2:		
	1.000000	0.
	1.000000	0.
	1.000000	0.
EIGENVALUE 3:	1.000000	1.000000
EIGENVECTØR 3:		
	0.000000	0.000000
	1.000000	0.
	0.000000	0.000000

```
ENTER ØRDER ØF MATRIX
=0
```

This Fortran subroutine finds the eigenvalues and eigenvectors of a real, nonsymmetric matrix. It also has the capability to determine a specified subset of the eigenvectors.

INSTRUCTIONS

The calling sequence for EIGNSR is:

CALL EIGNSR (A, NC, NV, ROOT, VECT, TEMP1, TEMP2, ITEMP3, IDIMA)

where

- A is the real matrix containing the eigenvalues and eigenvectors.
- NC is the order of the matrix, i.e., A is a NC by NC matrix.
- NV determines which vectors are found. If
  a. NV.GT.O, finds the eigenvectors associated with the NV eigenvalues of largest modules.
  b. NV.LT.O, finds the eigenvectors associated with the IABS(NV) eigenvalues of smallest modules.
  c. NV=O, no eigenvectors found.
- ROOT is the 1-dimensional complex array of the eigenvalues.
- VECT is the 2-dimensional complex array of the eigenvectors. The eigenvector corresponding to the I^{th} eigenvalue is stored columnwise in the I^{th} column of VECT.
- TEMP1 is a 2-dimensional complex array used internally. It is dimensioned as follows: TEMP1(IDIMA, NC).
- TEMP2 is a 2-dimensional complex array used internally. It is dimensioned as follows: TEMP2(IDIMA, 2).
- ITEMP3 is a 2-dimensional integer array used internally. It is dimensioned as follows: ITEMP3(IDIMA, 2).
- IDIMA is the first dimension of the following arrays: A, VECT, TEMP1, TEMP2, ITEMP3.

RESTRICTIONS AND METHOD

1. After control is returned to the main program following a call to EIGNSR, the routine may be entered again to find the eigenvectors of the transpose of A. This option must be used carefully, with the following restrictions:
   a. The transpose of A must be stored before the first call to EIGNSR since the routine destroys A.
   b. The array ROOT must be the same in both calls to the routine since it contains the eigenvalues of A and, therefore, of A transpose; because of this, the second call to the routine does not recompute eigenvalues of A transpose. Therefore, no change can be made to the ROOT array before the second entry into EIGNSR.

c. The reentry to the routine must be made with a minus NC in place of NC. This informs the routine that it already has the eigenvalues of A transpose which are stored in ROOT.

d. Upon reentering, NV need not be the same as it was in the original entry.

This option provides the capability of solving the complete eigen problem, assuming the Jordan form D of A is diagonal. If X designates the matrix whose columns are eigenvectors of A, and Y designates the matrix whose columns are eigenvectors of A transpose, with proper scaling of X and Y, the following relations hold:

AX =XD (AT = A transpose, YT=Y transpose)

(AT)Y =YD

(YT)X =I (Scaling required to produce unit diagonal elements)

(YT)AX =D (The diagonal matrix D has the eigenvalues on the diagonal.)

2. The eigenvectors returned by EIGNSR are normalized so that the largest component is 1.

3. It is suggested that the user check the modulus of the first component of each eigenvector after a call to EIGNSR. If the modulus is equal to 2., the routine has failed to determine the particular eigenvector. To determine the modulus use the CABS function.

REFERENCES

1. Francis, J.G.F., "The QR Transformation, A unitary Analog to the LR Transformation," Part I, Computer Journal, Oct., 1961, 265-271.

2. _____, Part II, Computer Journal, Jan., 1962, 332-345.

3. Wilkinson, J.H., The Algebraic Eigenvalue Problem, Oxford Press, 1965.

4. G. E. TIS #67SD335, Eigenvalues and Eigenvectors of Non-Hermitian Matrices, by F. E. Lilley and A. T. Ross, G.E. Technical Information Exchange, P.O. Box 43, Bldg. 5; Schenectady, N.Y. 12301

SAMPLE PROBLEM

Given the following matrices

$$A=\begin{bmatrix} 1. & -1. & -1. \\ 1. & -1. & 0. \\ 1. & 0. & -1. \end{bmatrix}, \quad B=\begin{bmatrix} -2. & -8. & -12. \\ 1. & 4. & 4. \\ 0. & 0. & 1. \end{bmatrix} \quad C=\begin{bmatrix} 3. & 2. & 2. & -4. \\ 2. & 3. & 2. & -1. \\ 1. & 1. & 2. & -1. \\ 2. & 2. & 2. & -1. \end{bmatrix}$$

find all eigenvalues and eigenvectors of A and A transpose, all eigenvalues of B, and the two largest eigenvalues and their associated eigenvectors of C and C transpose.

SAMPLE SOLUTION

```
10        PARAMETER MAX=10
20        DIMENSION A(MAX,MAX),AA(MAX,MAX),ITEMP3(MAX,2)
30        COMPLEX TEMP1(MAX,MAX),ROOT(MAX),TEMP2(MAX,2),VECT(MAX,MAX)
40        COMPLEX VECT2(MAX,MAX)
50      1 WRITE(6,1006)
60        READ , NC
70        IF(NC .LE. 0) STOP
80        WRITE(6,1007)
90        READ , NV
100         WRITE(6,1008)
110       READ , ((A(I,J),J=1,NC),I=1,NC)
120       DO 10 I=1,NC
130       DO 10 J=1,NC
140    10 AA(J,I)=A(I,J)
150       CALL EIGNSR(A,NC,NV,ROOT,VECT,TEMP1,TEMP2,ITEMP3,MAX)
160       IF(NV)15,50,15
170    15 CALL EIGNSR(AA,-NC,NV,ROOT,VECT2,TEMP1,TEMP2,ITEMP3,MAX)
180       WRITE(6,1004)
190       IF(NV)20,50,25
200    20 K=NC+NV+1
210       KK=NC
220       GO TO 30
230    25 K=1
240       KK=NV
250    30 DO 40 I=K,KK
260       WRITE(6,1002)I,ROOT(I)
270       WRITE(6,1009)I
280    40 WRITE(6,1003)(VECT(J,I),J=1,NC)
290       WRITE(6,1005)
300       DO 45 I=K,KK
310       WRITE(6,1002)I,ROOT(I)
320       WRITE(6,1009)I
330    45 WRITE(6,1003)(VECT2(J,I),J=1,NC)
340       GO TO 1
350    50 DO 60 I=1,NC
360    60 WRITE(6,1002)I,ROOT(I)
370       GO TO 1
380  1002 FORMAT(/' EIGENVALUE',I3,':',2F12.6)
390  1003 FORMAT(15X,2F12.6)
400  1004 FORMAT(////25X,'MATRIX'/20X,'REAL',6X,'IMAGINARY')
410  1005 FORMAT(////18X,'MATRIX TRANSPOSE'/20X,'REAL',6X,'IMAGINARY')
420  1006 FORMAT(///'ENTER ORDER OF MATRIX')
430  1007 FORMAT(/'ENTER NV CODE')
440  1008 FORMAT(/'ENTER MATRIX BY ROWS')
450  1009 FORMAT(' EIGENVECTOR',I3,':')
460       END
```

READY

```
*RUN *;EIGNSR=(CORE=25)


ENTER ORDER OF MATRIX
=3

ENTER NV CODE
=3

ENTER MATRIX BY ROWS
=1, -1, -1
=1, -1,  0
=1,  0, -1
```

MATRIX

	REAL	IMAGINARY
EIGENVALUE 1:	-0.000000	-1.000000
EIGENVECTOR 1:		
	1.000000	0.
	0.500000	0.500000
	0.500000	0.500000
EIGENVALUE 2:	-0.000000	1.000000
EIGENVECTOR 2:		
	1.000000	0.
	0.500000	-0.500000
	0.500000	-0.500000
EIGENVALUE 3:	-1.000000	0.
EIGENVECTOR 3:		
	0.000000	0.
	1.000000	0.
	-1.000000	0.

MATRIX TRANSPOSE

	REAL	IMAGINARY
EIGENVALUE 1:	-0.000000	-1.000000
EIGENVECTOR 1:		
	1.000000	0.
	-0.500000	-0.500000
	-0.500000	-0.500000
EIGENVALUE 2:	-0.000000	1.000000
EIGENVECTOR 2:		
	1.000000	0.
	-0.500000	0.500000
	-0.500000	0.500000
EIGENVALUE 3:	-1.000000	0.
EIGENVECTOR 3:		
	-0.000000	0.
	1.000000	0.
	-1.000000	0.

```
ENTER ØRDER Ø* MATRIX
=3

ENTER NV CØDE
=0

ENTER MATRIX BY RØWS
=-2, -8, -12
= 1,  4,   4
= 0,  0,   1

EIGENVALUE  1:     0.              0.

EIGENVALUE  2:     2.000000        0.

EIGENVALUE  3:     1.000000        0.

ENTER ØRDER ØF MATRIX
=4

ENTER NV CØDE
=2
ENTER MATRIX BY RØWS
=3, 2, 2, -4
=2, 3, 2, -1
=1, 1, 2,-1
=2, 2, 2, -1

                        MATRIX
                      REAL          IMAGINARY

EIGENVALUE  1:     3.000000        0.
EIGENVECTØR  1:
                   0.000000        0.
                   1.000000        0.
                   0.333333        0.
                   0.666667        0.

EIGENVALUE  2:     2.000000        0.
EIGENVECTØR  2:
                  -0.500000        0.
                   1.000000        0.
                   0.250000        0.
                   0.500000        0.
                  MATRIX TRANSPØSE
                    REAL          IMAGINARY

EIGENVALUE  1:     3.000000        0.
EIGENVECTØR  1:
                  -0.666667        0.
                  -0.666667        0.
                  -0.666667        0.
                   1.000000        0.

EIGENVALUE  2:     2.000000        0.
EIGENVECTØR  2:
                  -0.500000        0.
                  -0.500000        0.
                  -0.500000        0.
                   1.000000        0.

ENTER ØRDER ØF MATRIX
=0

NØRMAL TERMINATION

*
```

This Fortran subroutine solves a system of simultaneous linear equations with symmetric double-precision coefficient matrix. Advantage is taken of the symmetry to save time and storage.

INSTRUCTIONS

The calling sequence is:

CALL LINSD (A, B, NA, NB, EPS, IER, AUX, IDIM)

where

- A is the name of a double-precision, single-dimension array in which the upper triangle of the coefficient matrix is stored columnwise in NA*(NA+1)/2 successive locations. A is destroyed by the subroutine.
- B is the name of a double-precision, double-dimension NA by NB array containing the right-hand side vectors. On return, B contains the solution vectors.
- NA is the number of equations in the system.
- NB is the number of right-hand side vectors.
- EPS is a single-precision criterion for determining possible loss of significance.
- IER is an error return as follows:

IER=0	indicates no error.
IER=-1	indicates no result because NA was less than 1, or a pivot element was equal to 0 during elimination, indicating A may be singular.
IER=K	is a warning of possible loss of significance at elimination step K+1. Calculations are continued.

- AUX is a double-precision auxiliary storage array with dimension NA-1.
- IDIM is the first dimension of B assigned by the dimension statement of the main program.

METHOD

1. Gaussian Elimination is used with pivoting in the main diagonal only, to preserve symmetry.
2. An error return of IER=K indicates that at elimination step K+1, the absolute value of pivot element < AM*EPS, where AM equals the maximum absolute value of the main diagonal elements of A. If $EPS=10^{-L}$, a return of IER=K may be interpreted as indicating a possible loss of L significant digits at elimination step K+1, and, with well-conditioned A and appropriate EPS, that A may have a rank of K. A relative tolerance of 10^{-14} to 10^{-16} is suggested.

SAMPLE PROBLEM:

Solve the linear system AX=1, where I is the identity matrix and

$$A = \begin{pmatrix} 26. & 8. & 9. & 0. & 0. \\ 8. & 40. & 6. & 24. & 0. \\ 9. & 6. & 14. & 18. & 10. \\ 0. & 24. & 18. & 65. & 35. \\ 0. & 0. & 10. & 35. & 25. \end{pmatrix}$$

SAMPLE SOLUTION

```
010       DØUBLE PRECISIØN A(15), B(5,5), AUX(4)
020       DATA A /26.D0,   8.D0, 40.D0,   9.D0, 6.D0, 14.D0,
030&             0.D0, 24.D0, 18.D0, 65.D0,
035&             0.D0, 0.D0, 10.D0, 35.D0, 25.D0/
040       DATA B /25 * 0.0D0/
050       DØ 10 I=1,5
060    10 B(I,I) = 1.0D0
070       CALL LINSD (A, B, 5, 5, 1.D-15, IER, AUX, 5)
080       WRITE (6, 102) IER
090   102 FØRMAT (' IER = ', I5)
100       WRITE (6, 111) ((B(I,J),J=1,5), I=1,5)
110   111 FØRMAT ('SØLUTIØN MATRIX IS' / (5G14.6))
120       STØP
130       END

READY

*RUN *$LINSD=(CØRE=18K)
IER =      0
SØLUTIØN MATRIX IS
      1.000000       -2.00000       -1.00000        3.25000       -4.15000
     -2.00000         4.25000        2.00000       -6.87500        8.82500
     -1.00000         2.00000        1.11111       -3.27778        4.14444
      3.25000        -6.87500       -3.27778        11.1944       -14.3611
     -4.15000         8.82500        4.14444       -14.3611        18.4878

NØRMAL TERMINATIØN

*
```

This Fortran subroutine solves a system of simultaneous linear equations with symmetric single-precision coefficient matrix. Advantage is taken of the symmetry to save time and storage.

INSTRUCTIONS

The calling sequence is:

CALL LINSS (A, B, NA, NB, EPS, IER, AUX, IDIM)

where

- A is the name of a single-precision, single-dimension array in which the upper triangle of the coefficient matrix is stored columnwise in NA*(NA+1)/2 successive locations. A is destroyed by the subroutine.
- B is the name of a single-precision, double-dimension NA by NB array containing the right-hand side vectors. On return, B contains the solution vectors.
- NA is the number of equations in the system.
- NB is the number of right-hand side vectors.
- EPS is a single-precision criterion for determining possible loss of significance.
- IER is an error return as follows:

  IER = 0 indicates no error.

  IER = -1 indicates no result because NA was less than 1, or a pivot element was equal to 0 during elimination, indicating A may be singular.

  IER = K is a warning of possible loss of significance at elimination step K+1. Calculations are continued.
- AUX is a single-precision auxiliary storage array with dimension NA-1.
- IDIM is the first dimension of B assigned by the dimension statement of the main program.

METHOD

1. Gaussian Elimination is used with pivoting in the main diagonal only, to preserve symmetry.
2. An error return of IER=K indicates that at elimination step K+1, the absolute value of pivot element < AM*EPS, where AM equals the maximum absolute value of the main diagonal elements of A. If $EPS=10^{-L}$, a return of IER=K may be interpreted as indicating a possible loss of L significant digits at elimination step K+1, and, with well-conditioned A and appropriate EPS, that A may have a rank of K. A relative tolerance of 10^{-6} to 10^{-7} is suggested.

SAMPLE PROBLEM

Solve the linear system AX=1, where I is the identity matrix and

$$A = \begin{pmatrix} 26. & 8. & 9. & 0. & 0. \\ 8. & 40. & 6. & 24. & 0. \\ 9. & 6. & 14. & 18. & 10. \\ 0. & 24. & 18. & 65. & 35. \\ 0. & 0. & 10. & 35. & 25. \end{pmatrix}$$

SAMPLE SOLUTION

```
010       DIMENSION A(15), B(5,5), AUX(4)
020       DATA A /26.,   8., 40.,   9., 6., 14.,
030&             0., 24., 18., 65.,   0., 0., 10., 35., 25./
040       DATA B /25 * 0.0/
050       DO 10 I=1,5
060    10 B(I,I) = 1.0
070       CALL LINSS (A, B, 5, 5, 1.E-7, IER, AUX, 5)
080       WRITE (6, 102) IER
090   102 FORMAT (' IER = ', I5)
100       WRITE (6, 111) ((B(I,J),J=1,5), I=1,5)
110   111 FORMAT ('SOLUTION MATRIX IS' / (5G14.5))
120       STOP
130       END
```

```
READY

*RUN *,LINSS=(CORE=18K)
IER =      0
SOLUTION MATRIX IS
      1.00000       -2.0000       -1.00000        3.2500       -4.1500
     -2.0000         4.2500        2.0000        -6.8750        8.8250
     -1.00000        2.0000        1.1111        -3.2778        4.1444
      3.2500        -6.8750       -3.2778        11.194        -14.361
     -4.1500         8.8250        4.1444       -14.361         18.488

NORMAL TERMINATION

*
```

This Fortran subroutine transposes a matrix. The transposed matrix may be stored in either a distinct array or the same array as the original matrix.

INSTRUCTIONS

The calling sequence for this routine is:

CALL MTRAN (IR, IC, A, IA, B, IB)

where

IR is the number of rows of input matrix (number of columns of output matrix)

IC is the number of columns of input matrix (number of rows of output matrix)

A is the input matrix

IA is the row (first) dimension of A

B is the output matrix

IB is the row (first) dimension of B

The A and B arrays may be either the same array or distinct arrays.

SAMPLE PROBLEM

Transpose the matrix $\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$

SAMPLE SOLUTION

```
*10 DIMENSION A(5,6)
*20 DO 10 I=1,4
*30 A(1,I)=I
*40 10 A(2,I)=I+4
*50 PRINT,"ORIGINAL MATRIX:"
*60 PRINT 20, ((A(I,J),J=1,4),I=1,2)
*70 20 FORMAT(1X,4F3.0)
*80 CALL MTRAN(2,4,A,5,A,5)
*90 PRINT,"TRANSPOSED MATRIX:"
*100 PRINT 30, ((A(I,J),J=1,2),I=1,4)
*110 30 FORMAT(1X,2F3.0)
*120 STOP
*130 END
*RUN *;MTRAN
ORIGINAL MATRIX:
 1. 2. 3. 4.
 5. 6. 7. 8.
TRANSPOSED MATRIX:
 1. 5.
 2. 6.
 3. 7.
 4. 8.

NORMAL TERMINATION

*
```

This BASIC program integrates a function by spline fits, with the function defined by possibly unequally spaced data points.

INSTRUCTIONS

Starting with statement number 700, the first DATA statement must be the number of pairs of x and y. The DATA statements that follow must contain the values of x and y. After the data is entered, type RUN.

SAMPLE PROBLEM

Find the integral between 0 and 1 of the function described by the table below.

X	0	.1	.2	.3	.4	.5	.6	.7	.8	.9	1.0
Y	.0	.0398	.0793	.1179	.1554	.1915	.2257	.2580	.2881	.3159	.3413

SAMPLE SOLUTION

```
*700 DATA 11
*710 DATA 0,0
*720 DATA .1,.0398,  .2,.0793,  .3,.1179
*730 DATA .4,.1554,  .5,.1915,  .6,.2257
*740 DATA .7,.2580,  .8,.2881,  .9,.3159
*750 DATA 1.,.3413
*RUN

SPLINE

    X             Y             INTEGRAL

          0             0                0
         .1          .0398        .0019911
         .2          .0793        .0079511
         .3          .1179        .0178199
         .4          .1554        .0314948
         .5          .1915        .0488539
         .6          .2257          .06973
         .7           .258        .0939314
         .8          .2881        .1212573
         .9          .3159        .1514695
          1          .3413        .1843773

READY
*
```

This BASIC program computes spline interpolation.

INSTRUCTIONS

Starting in line 1900, the first DATA statement must be the number of pairs of x and y, followed by the number of interpolation points desired. The DATA statements that follow must contain the values of x and y, followed by the x values at which interpolations are to take place. After the data is entered, type RUN.

SAMPLE PROBLEM

Given the following table, use spline interpolation to find the values of y for x = .23, .57, .65

X	0	.1	.2	.3	.4	.5	.6	.7	.8	.9	1.0
Y	.0	.0398	.0793	.1179	.1554	.1915	.2257	.2580	.2881	.3159	.3413

SAMPLE SOLUTION:

```
*1900 DATA 11,3
*2000 DATA 0,0,  .1,.0398,  .2,.0793,  .3,.1179
*2010 DATA .4,.1554,  .5,.1915,  .6,.2257,  .7,.2580
*2020 DATA .8,.2881, .9,.3159,  1.,.3413
*2100 DATA .23, .57, .65
*RUN


SPLINT

10 OF POINTS GIVEN =    11
NO OF INTERPOLATED POINTS =     3

INTERP X        INTERP Y

          .23         .0909903
          .57         .2156386
          .65         .2421052

ORIGINAL X      ORIGINAL Y

            0               0
           .1            .0398
           .2            .0793
           .3            .1179
           .4            .1554
           .5            .1915
           .6            .2257
           .7             .258
           .8            .2881
           .9            .3159
            1            .3413

READY
*
```